Blockchain Framework

28 February 2018. Version 1.1

Table of Contents

Disclaimer

IMPORTANT INFORMATION

General

This White Paper provides a general technical overview of the proposed smart contract-based token generating event described in this White Paper (the "TGE") and, more generally, the Plug Platform. This White Paper is for information purposes only.

None of Plug Global Limited, Plug International Limited, any of their associated entities, nor any of their employees, agents or assigns (together, the "Plug Entities", each a "Plug Entity") make any warranty or representation (to the extent permitted by law) as to the utility or value of PLUG tokens or the operation of the Plug Platform.

Not a regulated offer of financial products

This White Paper, and the information contained in it and any information accompanying it, are not, and are under no circumstances to be construed as, an offer of financial products to any person who requires disclosure under the Financial Services and Markets Act 2000 or any other securities law. This White Paper is not a product disclosure statement and does not contain all the information that a product disclosure statement is required to contain under English law. Neither this White Paper, nor any other document, have been registered, filed with or reviewed or approved by any English regulatory authority or under or in accordance with the FMC Act.

In addition, this White Paper, and the information contained in it and any information accompanying it, are not, and are under no circumstances to be construed as an offer to sell or issue, or a solicitation of an offer to purchase or subscribe for, Regulated Products subject to regulation by the FCA or PRA as set out in the Financial Services and Markets Act 2000 (Regulated Activities) Order 2001, SI 2001/544.

Not a retail securities offer

Without limiting the generality of the preceding paragraph, this White Paper (including the information contained in it and any information accompanying it) is not, and is under no circumstances to be construed as an offer to sell or issue, or a solicitation of an offer to purchase or subscribe for, any securities to any retail investor in any jurisdiction.

Wholesale investor warranties and selling restrictions

Any person who applies to buy PLUG tokens in the TGE irrevocably and unconditionally represents and agrees (among other matters that are set out in the subscription agreement) that:

(a) the person is not a citizen or a resident of the United States of America, New Zealand or of the People's Republic of China, nor a citizen or a resident of any country or jurisdiction in which the sale, purchase, disposal, acquisition, holding and/or trading of tokens issued

pursuant to a TGE is not permitted by law, nor is the person located in a geographic area that is subject to the laws of any such country;

(b) the person is not a citizen or a resident of, or is physically located in, the Republic of Singapore;

(c) the person is a "wholesale investor" (or the equivalent term) for the purposes of the securities laws in all applicable jurisdictions;

(d) the person is not purchasing the PLUG tokens as agent or otherwise for or on behalf of any other person or entity; and

(e) the person has not:

(i) offered or sold, and will not offer or sell, directly or indirectly, any PLUG token prior to the date on which Plug Global Limited, which will sell and issue the PLUG tokens, notifies purchasers that the Plug Platform has been released which will not occur until at least 3 months after the end of the TGE (the "Release Date"); and

(ii) distributed and will not distribute, directly or indirectly prior to the Release Date, the White Paper, Plug Global Limited's constitution, or any other offering materials or advertisements in relation to the TGE or the PLUG tokens,

other than to persons who meet all of the criteria set out in paragraphs (a) to (d) above, and where doing so will not result in (i) the sale of the PLUG tokens in the TGE being, or being viewed as, a regulated offer of financial products or securities in any country or jurisdiction, (ii) any contravention by any Plug Entity or any other persons of FSMA or any other applicable laws or regulations in any country or jurisdiction or (iii) the seller and issuer of the PLUG tokens, Plug Global Limited, or other Plug Entities, or any of their respective directors and officers, incurring any liability in any country or jurisdiction.

Anti-money laundering procedures

Any person who applies to purchase PLUG tokens in the TGE will be subject to strict anti-money laundering and countering the financing of terrorism checks. These are detailed in the subscription agreement.

Limited role of Plug International Limited

Plug International Limited is acting solely as an arms' length third party organizer of the TGE and not in the capacity as the financial adviser or fiduciary of any person with regard to the sale of PLUG tokens.

No guarantee

Neither Plug International Limited nor any other person guarantees the performance of the Plug Platform, the obligations of Plug Global Limited, nor the functionality of the PLUG tokens.

No other material authorised

Save for persons who are expressly authorised by Plug Global Limited in writing, no person is authorised to give any information or to make any representation relating to the PLUG tokens outside of this White Paper and the subscription agreement.

No offer or contract

This White Paper is not an offer or invitation to any person to purchase or otherwise deal in any PLUG tokens. It does not create a contract between the Plug Global Limited and/or Plug International Limited with any person.

No recommendation or financial advice

This White Paper has been prepared solely for general information purposes and not as specific advice to any particular prospective purchaser of PLUG tokens. It is not an express or implied recommendation to any person to purchase PLUG tokens. In addition, any information or documentation (including this White Paper and the subscription agreement) provided to you by Plug Global Limited and/or Plug International Limited (or any of their affiliates), or anyone acting on their behalf, does not purport to be, and shall not in any way be construed as constituting, the provision of any advice or recommendation relating to any Regulated Product nor constitute any financial, business, accounting, legal or tax advice or recommendation.

Limitation of liability

Plug Global Limited had no involvement with the 'Early Contribution Round' for PLUG Round A tokens arranged by Plug International Limited. Plug Global Limited and its directors will bear no responsibility or liability whatsoever for any loss or liability incurred by any person relating to the PLUG Round A tokens. Plug Global Limited shall not have any liability in respect of any claim by any person (whether under law or otherwise) for any loss of business or profits, or in connection with any indirect or consequential loss or any punitive or aggravated damages, arising out of any matter or circumstances giving rise to a claim (whether under law or otherwise).

Risk summary

The purchase of PLUG Tokens involves considerable risk. These risks are described in the 'Risks Summary' section of this White Paper. Tax Holders that acquire PLUG tokens are wholly responsible for understanding and meeting all their tax obligations in relation to their acquisition, holding or disposal of PLUG tokens.

Any payments that are made by Plug Global Limited to any holder of PLUG tokens will be made after the deduction of any withholding taxes, if so applicable.

Reliance

No person is entitled to rely on the contents of this White Paper or any inferences drawn from it. This White Paper is not intended to substitute or replace the due diligence that a purchaser should undertake before deciding whether or not to buy PLUG tokens. The information in

this White Paper is not intended to be relied upon in relation to, and must not be taken as basis for, any decision to subscribe for PLUG tokens. Each recipient of this White Paper is to rely solely on their own knowledge, investigation judgment and assessment of the matters which are the subject of this White Paper and any information made available in connection with further enquiries. A person must satisfy itself as to the accuracy and completeness of the information contained within this White Paper before entering into a subscription agreement for the subscription of PLUG tokens.

While the Plug Entities have acted in good faith and have made every effort to ensure that the statements made in this White Paper are reliable and accurate, and that all estimates, forecasts, expressions of opinion and other subjective judgments contained in this White Paper are based on assumptions considered to be reasonable as of the date of this White Paper, no warranty or guarantee, or representation (whether written, oral or otherwise) is made by the Plug Entities with regard to the accuracy, completeness or suitability of the information presented in this White Paper (to the extent permitted by law). This White Paper should not be relied upon, and shall not confer rights or remedies upon you or any other person. To the maximum extent permitted by law, the Plug Entities have no obligation to amend, modify or update this White Paper should any of the information presented herein change or subsequently become inaccurate, incomplete or unsuitable.

White Paper may be updated from time to time

This version of the White Paper may be updated from time to time. You should ensure you read the current version available here

By accessing this White Paper, you irrevocably and unconditionally agree to be bound by all of the above terms.

Introduction & The Plug Vision

PLUG, from a product perspective, has built from scratch a platform for building blockchain-based applications and services. PLUG has not been built on Bitcoin, Ethereum, or any other cryptocurrency code base; instead, we have built PLUG to directly address the needs of business-driven blockchains.

The inspiration behind the first version of PLUG was seeing people trying to create interesting, especially non-financial, use cases on top of the Bitcoin codebase in late 2013 and having a difficult time due to the inflexible nature of the Bitcoin architecture.

Combining some of the best ideas from Bitcoin, along with research from traditional consensus and distributed systems, it soon became apparent how to provide similar, underlying functionality in a more extensible way. This led to the first version of PLUG, and the philosophy that security at the

platform level does not have to get in the way of usability at the service level has guided our company's path every step of the way.

Our central thesis is that, hype aside, the central value proposition blockchain will provide in practice is the movement of certain types of organizational trust and management of identity from a manual, operationally-driven process to an automated, protocol-driven process.

This is similar to the way 'The Cloud' was overhyped early in its lifecycle as the be-all and end-all to all things corporate infrastructure, but in the end the impact could be summed up as giving organizations the ability to decide whether they would deploy infrastructure in a way that optimized for CapEx or OpEx1.

Learning from the way these earlier technologies matured from over-hyped, breathless enthusiasm into boring workhorses that power today's infrastructure, we have attempted from the start to build PLUG as a truly horizontal, general-purpose technology layer with the ability to integrate natively with other, existing layers of technology.

Just as you do not need to know how to build a web server from scratch to build a website, we have ensured you do not need to know how to build a blockchain from scratch to build a blockchain.

You use and integrate our framework much like any other traditional software library, removing much of the execution and maintenance risks associated with competing blockchain frameworks.

The systems we do have that rely on real-world identity are generally expensive to onboard to, necessitate a hub-and-spoke authority, and are (usually) cordoned off to interactions with either one organization or one use case. With blockchain, we see the potential in changing this across a wide range of industries.

Through the rest of this paper, we will lay out how blockchain came about as a concept in the first place, how we made both the technical and non-technical decisions that guided our platform, and what use cases we believe this technology will have a positive impact upon.

As always, the best way of learning is through experience. If the ideas presented here seem practical as applied to your field or domain, it's likely there's a good fit on the technology side as well, and we are more than happy to work with you to evaluate if that is the case.

Blockchain – From the Beginning

While the word "blockchain" made its first appearance in conjunction with Bitcoin, the underlying concepts and building blocks have been the subject of both research and practical use cases for many years previously.

---

[1] Other technologies that also fit this pattern include 'Mobile', 'SaaS', 'Big Data', and 'NoSQL'.

Much like most previously hyped technologies, such as mobile and cloud computing, blockchain is largely an assembly of several, complementary core technologies and becomes practical due to each of the core technologies progressing to a "good enough" threshold to be used in practice; in this case cryptography, consensus, distributed systems, and tamper-proof hardware.

Predecessors to Bitcoin

Bitcoin itself was a slight twist on Adam Back's HashCash[2] protocol, an idea announced in 1997 for combatting email spam by attaching a special header that proved the sender has performed a certain amount of computation before sending the email; HashCash was, in fact, used early on by SpamAssassin and Microsoft to attempt to accomplish just this.

It was also recognized as a potential way of bootstrapping a currency by various people. For example; Nick Szabo, Hal Finney, and Wei Dai all proposed projects that looked similar in the way they utilized the HashCash Proof of Work mining and a way of assigning tokens in a cryptographically-based e-money system.

Even earlier than this, we saw David Chaum's ecash[3] protocol published in 1983, laying out a practical, scalable solution for centrally-issued tokens that can be used truly anonymously. In 1990, they proposed a slight modification that even allowed this payments system to be used offline. This was used in production by a single bank in the US for three years but never reached widespread adoption due to a combination of market and personality issues.

Cypherpunks, Cyberpunks, and Libertarians

Of course, the existence of a technology, new or otherwise, does not guarantee its adoption, hype, or usefulness. In the case of these proto-blockchain technologies, there was often an academic or ideological reason for exploring the initial technology, but often not a market reason or a sufficient user base that was able to kick-start a sufficient critical mass for mainstream acceptance.

This was largely the case for the technologies discussed in the previous section. They were still waiting for the right combination of narrative and initial community to light the initial spark for sufficient excitement and usage to emerge and propel these technologies into the limelight.

Fortunately for Bitcoin, three previously marginalized groups found the early internet of the late 90s and early 2000s to be a great feeding ground for building their respective communities: The Cypherpunks, Cyberpunks, and Libertarians:

---

2 https://en.wikipedia.org/wiki/Hashcash
3 https://en.wikipedia.org/wiki/Ecash

- **Cypherpunks** – Largely speaking, Cypherpunks are activists who advocate the widespread use of strong cryptography (writing in code) as a route to progressive change[4]
- **Cyberpunks** – A subgenre of science fiction in a future setting that tends to focus on society as "high tech low life" featuring advanced technological and scientific achievements, such as information technology and cybernetics, juxtaposed with a degree of breakdown or radical change in the social order.[5]
- **Libertarians** – A political group that shares a scepticism of authority and state power. Believes in moving many government functions, often including defence, policing, and money to the private sector.[6]

It's from these three groups that the initial support base for Bitcoin arose, and it's from these roots that you see many of the trends within Bitcoin stem.

The Cypherpunks were largely the drivers and main supporters of the technologies discussed in the last section. It's from this group that we also first derived technologies such as PGP/GPG, but also where we have historically seen technologies that cannot scale beyond an initial, dedicated group of participants[7][8].

The Cypherpunks were, largely speaking, the groups that were primarily involved in the HashCash-based pre-Bitcoin cryptocurrencies.

The Cyberpunks are generally a group more concentrating on the stylistic and fiction side of how cryptography and related information technologies may impact society.

Much like traditional Science Fiction, this can give us quite a good roadmap of the opportunities and challenges we may encounter as we start to incorporate this technology into society; however, we must keep in mind that, like many communities, this community is very polarized in how optimistic and/or pessimistic their takes on the potential of the future is, so the possibilities explored must be taken with quite a grain of salt.

The Libertarians are, as a rule, suspicious of authority and centrally-controlled structures. A large contingent has believed in gold-backed (or other commodity) money and is suspicious of fiat-issued money.

As a result, various privately-issued money providers (e-gold, Liberty Reserve, etc.) popped up to serve this demographic during the mid-2000s. These were largely either shut down due to KYC/AML concerns or were forced to pivot to add more stringent identity and money laundering controls.

---

[4] https://en.wikipedia.org/wiki/Cypherpunk
[5] https://en.wikipedia.org/wiki/Cyberpunk
[6] https://en.wikipedia.org/wiki/Libertarianism
[7] https://moxie.org/blog/gpg-and-me/
[8] https://lists.torproject.org/pipermail/tor-talk/2013-September/030235.html

This issue came to a head in these communities following the 2008 crisis where many pinned most of the underlying causes of the crisis (and most financial crises) to the entire central banking system and strongly believed that a private currency (or group of private currencies) could avoid similar outcomes going forward.

After Bitcoin's official launch in early 2009[9], several of these groups latched onto the fledgling network and, through the first several years of Bitcoin's life, provided much of the initial community, incoming funding, and press that Bitcoin initially enjoyed.

As Bitcoin started reaching highs of over $1,000 per Bitcoin in late 2013, these three groups became exposed to various mainstream audiences and platforms, and the hype cycle of blockchain and its potential to disrupt everything from finance to government began.

Consensus, PoW, & PoS

Blockchains, at their core, are a method of reaching distributed history via a consensus mechanism. Bitcoin is still the most prominent example in the mainstream mind as an example, and it uses Proof of Work (PoW) mining as its consensus mechanism; however, there are many consensus mechanisms that have been developed in academia and industry over the previous several decades, and many of them are completely valid for use in building a blockchain.
It turns out, ourselves and others have discovered[10] that the only real benefit to PoW mining is that it allows arbitrary, anonymous validators to contribute to the network, without requiring an explicit onboarding process.

However, this comes at great expense to computational power, loss of control over your network, loss of flexibility of use case, and loss of scalability.
To avoid these limitations in the enterprise, many traditional consensus algorithms have been brought off the shelf, dusted off, and are now finding new life as the basis for blockchain networks.

Many of these algorithms have much nicer properties in scalability, flexibility, and ability to administrate, and the only real trade-off vs PoW is that participants must be at least pseudonymous; of course, every single block in Bitcoin for the past several years has been signed with the real-world identity of the entity who mined it anyway, so ironically Bitcoin is not utilizing the only functional advantage their consensus mechanism has over all the others.

---

[9] https://en.bitcoin.it/wiki/Genesis_block - There is evidence through much of his early involvement in the community that Satoshi Nakamoto was strongly against fiat currency and the 2008 bailouts, including a reference to them in the initial (or Genesis) block of Bitcoin.

[10] http://PLUG.vision/posts/permissionless-vs-permissioned-consensus-tradeoffs

We will ignore for a moment all the problems PoW mining is seeing in practice vs the vision laid out in the Satoshi whitepaper, but suffice it to say that precious few of the claims regarding scalability, functionality, and even decentralization have stood up to muster in the previous 8 years.[11]

Blockchain Isn't Magic

The first practical lesson of blockchain is that blockchain does not fundamentally remove intermediaries; blockchains themselves are intermediaries. They are intermediaries that may be owned by distributed groups of individuals, but they are the medium through which one transacts, they carry fees, and you're reliant on their abilities to process your transactions.
If it walks like a duck, and it talks like a duck, it may be a duck; even if you assign agency to each wing, bill, and breast.

In the words of Dave Birch, blockchain isn't just magic pixie dust you can sprinkle on society's problems and expect them to disappear.
To be fair, the reason so many of these use cases have been highlighted as potential use cases for blockchain is indeed exactly because they're the use cases where it is expensive or friction-filled to engage in those use cases today; but these are hard problems because they're actually hard problems of managing trust and relationships.

Incredibly smart people have been working on solving these problems for millennia, and not because we were waiting on One Weird Trick! Bankers Hate Her![12]

Many in the cryptocurrency community act like the problem is that we're relying on trust, and they can kill the werewolf of trust with a single, silver bullet of blockchain. However, in the real world, silver bullets are expensive and ineffective; the actual way to move forward is through diligent and accurate firing of lead bullets.
Look for those who are settling the frontier, rather than those convinced they've wandered into Grimms' Fairy Tales.

Removal of Trust vs Management of Trust

PLUG very much views blockchain as a mechanism for better identifying, onboarding, and managing trust in systems which have potentially untrusted participants.

Rather than a mechanism for removing trust, we see parallels to the Single Responsibility Principle in Computer Science[13], which states that each component of a software system should take full

---

[11] http://PLUG.vision/posts/trust-not-transactions-1-federation-not-decentralization
[12] http://www.slate.com/articles/business/moneybox/2013/07/how_one_weird_trick_conquered_the_internet_what_happens_when_you_click_on.html
[13] https://en.wikipedia.org/wiki/Single_responsibility_principle

responsibility for a specific part of functionality, and that responsibility should be impossible for any other part of the system to sabotage. We see a parallel in the way that many organizations with a fiduciary duty are set up[14] and seek to model that in software natively.

Trust as the Bedrock of Civilization

One of the worst things about the word 'trust' (much like the word 'identity') is that it often acts as a short-circuit for thought as people hear the word, and it feels like such a simple concept that almost any plausible-sounding argument that follows is accepted unless the person has in-depth domain knowledge of the topic at hand.

Instead, it's often worth diving into the mechanics of what 'trust' means in a given context, and evaluating whether a system meets the requirements of that situation.

Just like security, trust is not a binary, on/off state. Rather, we feel a system is 'trusted' if we can rely on the probability of a bad actor being able to inflict some negative outcome on us to be small enough that the benefits of using the system outweigh the risks.

Note that this is separate from whether a bad actor is inherently unable to inflict a negative outcome and merely the probability they will do so; an important distinction.

For example, we do precious little to prevent any random driver from running into any random pedestrian on most surface streets. While strategies like traffic calming and use of obstacles does much good in practice, we still have multi-ton vehicles zipping past large groups of pedestrians on a daily basis. However, we very rarely see someone refuse to go about their day because of fear that any passing driver could flick their wrist and end their life.

One of the reasons this works so well in the physical world is that most people are honest, most people don't want to break the law, and most people don't want to commit immoral actions. People, rightly or wrongly, feel accountable for their actions in the physical world and generally act accordingly.
In fact, it is generally understood that there is a direct trade-off between the levels of trust a society has and its costs to transact[15]

This also holds for individual businesses as it has been shown that individual businesses who have a high-trust culture tend to have outsize returns compared to businesses with a low-trust culture[16].
For further reading on this subject, Bruce Schneier's Liars and Outliers is a fantastic resource on how we manage trust on an individual and societal basis.

---

[14] For example, when you require your Financial Director to take two weeks off consecutively each year

[15] http://www.economist.com/news/special-report/21570835-nordic-countries-are-probably-best-governed-world-secret-their

[16] https://s3.amazonaws.com/media.greatplacetowork.com/pdfs/Business+Case+for+a+High-Trust+Culture_081816.pdf

This model often breaks down in the world of the internet, at least in its current implementation. As many interactions are anonymous or pseudonymous, and computers allow for interactions to scale much beyond the 1-to-1 interactions of the physical world, there are a few main differences in how we must approach trust on the internet:

- Less accountability (real & perceived) for interactions on the internet than face-to-face.
- Significantly less cost to participate fraudulently for each interaction than in-person.
- Significantly easier to scale fraudulent activity on a 1-to-many scale, e.g., email phishing scammers.
- Generally, a significantly greater cost, direct as well as opportunity, to resolve issues after the fact.
- Dichotomy between laws in the jurisdiction a fraudulent person resides versus the victim of fraud.
- Ability to reduce or eliminate the chance of tracing the attack back to a specific individual.

As a result, the institutions that we have traditionally relied upon to manage trust and risk in our economy have been very reluctant to expose themselves too greatly to these issues. Rather, they have exposed specific functionalities (online banking, online brokerages, electronic stock exchanges, etc.) that are more straightforward to manage and deliver without exposing the institutions themselves to an unacceptable level of institutional risk in the process.

In addition, they have often failed to take advantage of consumer-facing technology that could mitigate these (very real) risks, but the main impediment to bringing a Silicon-Valley-esque experience to the financial markets has been from a position of minimizing the above risks, rather than a deliberate plan to block the potential benefits that a truly internet-enabled financial system could offer.

Bitcoin and cryptocurrencies have attempted to participate in this world by claiming they are able to remove all trust from a system with untrusted participants.

Given all we have discussed above, it should be relatively trivial to conclude this is both impossible in the general sense, but also undesirable from the point of view that everyone succeeds more in a high-trust environment.

We should be seeing how technology can build a high-trust environment, based on how we build trust in the physical world, rather than promising we can remove the need for trust from society altogether.

If you look at the reality of what you're trusting when you're using Bitcoin, as the prime example of the cryptocurrency model, you're not even removing trust from an oligarchy of participants, you're simply shifting it to an oligarchy of 3 to 9 individuals, most of whom happen to live in China.

The elephant in the room that all of finance has been trying to address for millennia has always been trust.

The problem of Bitcoin is that instead of removing the elephant, they simply threw a rug on the top of it and then exclaimed, "Look at this nice, empty room we have!"

Cryptocurrency vs Blockchain/DLT in Industry

While the first parties to talk about private blockchains started discussing the concept in 2014, it proved to be quite the uphill battle to gain widespread acceptance that this was a concept worth seriously discussing.

On one side, you had Bitcoin maximalists who were generally quite dismissive there would ever be a blockchain that wasn't Bitcoin that was worth talking about. On the other side, you had the financial institutions (blockchain still hadn't made the leap to non-financial use cases yet!) dismissing Bitcoin as a toy that would likely never scale to mainstream acceptance.

That's if they even got past the dismissal that Bitcoin was largely just a way of either:
1) Paying for illegal items online, or
2) Evading capital controls in China.

From here we witnessed two years of debate that largely ended with very little overlap between those who were pushing cryptocurrency-based use cases and those who were looking at how to solve problems in the enterprise, whether B2B or B2C, with blockchain-derived technologies.

During this period, we would go into various financial institutions, government agencies, and industry associations, and we would see a slow, downward movement in how much time was necessary to spend in each meeting on "What is blockchain?", "Isn't this just Bitcoin?", "Why should we investigate this tech?", and many other sceptical questions that one should logically be asking of any potentially over-hyped technology.

It's during this time we started aggressively exploring non-financial use cases, especially those which are government related.
We were fortunate enough to work with our local partners in the Isle of Man Government to launch the world's first government service running on a blockchain[17].

It's important to remember this was at a time where the narrative around blockchain from the cryptocurrency crowd was that blockchain was going to remove the need for all governments, rather than a tool which might allow them to serve their citizens in a more transparent and auditable manner.

---

[17] http://www.coindesk.com/isle-of-man-trials-first-government-run-blockchain-project

Now that we're a few years into this debate, and we've seen many companies raise huge sums on this premise, it's hard to believe there was ever a debate that private (or permissioned) blockchains were a viable concept.

Onboarding & Managing Identity and Trust

At a core technology level, blockchain simply provides a series of signed statements that can never be revoked and can be verified by all authorized participants.

These statements are signed with a machine-readable, cryptographic identity, but we can use this as a foundation for bringing real-world identity (individual, organizational, or physical) into a format more usable in the digital world.

We do this by pairing real-world identity to the cryptographic identity that the blockchain understands. This is similar in nature to the way Certificate Authorities[18] (CA) issue digital certificates to individuals and organizations today, but operates in a fashion that allows for more real-time management and recycling of these credentials and identity when necessary, as well as providing irrevocable audit trails for each interaction recorded by the blockchain.
Once this real-world identity is onboarded, the onboarded parties can use this identity in any use cases that rely on that identity, whether those use cases are on the blockchain or not.

One-Off vs Reusable Identities

This onboarding process itself will be conceptually similar to the way we perform KYC checks today. Some organizations that want to verify the real-world identity of someone will perform checks of provided identifying information, either manually or in an automated fashion, and if that information passes their minimum standard for doing business with a customer, they will have successfully completed their KYC process.

The difference in how we see blockchain-enabled identity playing out is in two directions:

- Where do organizations source identifying information?
- Where does this information go once it's verified?

For sourcing of information, instead of relying on scans of official documents to be uploaded by the user, we have been building a method to 'tokenize' verified data to be reused, either intra-organization or inter-organization.
This technique allows us to both protect privacy of the user by not sharing more than the necessary pieces of data another organization may require, but also raises the standard of proof of identity

[18] https://en.wikipedia.org/wiki/Certificate_authority

quite substantially, as you're relying on a direct, explicit assertion of identity, rather than inferring it from a digital scan of a physical document.

This also allows for an additional monetization model for those organizations that already exist as the de facto suppliers of identity in today's world.

For example, utility companies are one of the biggest verifiers and issuers of proof of identity in today's world, as a side effect of their core business function.

However, the current method of presenting and verifying these proofs is indirect, manual, expensive[19], and filled with friction.

Instead, we posit such providers have an amazing opportunity to provide a cryptographically-verifiable API endpoint, allowing for full user opt-in and consent, allowing them to capture some of the value they are already creating while reducing the other side of the cost of verification by an order of magnitude or more.

Modeling the Real World

Of course, the whole point of this endeavor is to be able to perform some function that involves people, organizations, and items in the real world, but enable the interaction via computers or the internet.

As we have historically been bad at building systems that can reliably accomplish this, blockchain provides an exciting possibility to better manage these relationships in an increasingly internet-driven world.

As these identities and relationships are built in a largely decentralized manner in the real world (i.e., I don't have to go through a third party to sign a contract with you), it makes sense to use a decentralized or federated technology to onboard and manage those relationships. In fact, one of the primary reasons there is so much friction, lack of access, and prevalence of fraud in many of the current systems we use for managing trust and relationships is that we don't use software systems that model the decentralized world we all live in today.

The 'Law of Leaky Abstractions'[20] informs us why we run into these issues due to the mismatch between the way the real world functions and the way we try to build infrastructure to manage these functions electronically. If your building blocks for a system don't represent what you're trying to

---

[19] https://www.thomsonreuters.com/en/press-releases/2016/may/thomson-reuters-2016-know-your-customer-surveys.html - Note the cost of penalties for incomplete or incorrectly verified information dwarfs the direct cost of the manual onboarding process.

[20] https://www.joelonsoftware.com/2002/11/11/the-law-of-leaky-abstractions/

model, you will spend most of your time trying to paper over the fact that your system can only deal with a simplified version of reality at its core.

Tokenization & Encapsulation of Information

Of course, it's easy to get lost in the weeds of any particular use case and what delivering that use case may entail. However, if you look at the common, underlying theme of these use cases, they generally boil down to making an agreement between two or more parties into a more manageable form.

For example, shares of stock in a company are effectively a glorified abstraction over an agreement the directors and existing shareholders made to issue shares; the share certificate itself is just a useful way of allowing that agreement to be bought, sold, and executed.

Consequently, a Share transfer is just another agreement to destroy an old share certificate belonging to the old owner and create a new one in the name of the new owner.

In fact, most of the use cases blockchain has a lot of potential to address tend to boil down to two mechanical parts:

- Associate a real-world identity (individual, organizational, or physical) with an electronic identity (public/private key pair).
- Record statements made, contracts agreed, or transactions initiated by these identities and then provide a more convenient and seamless way for managing these interactions.

The reason Blockchain is great at addressing these type of use cases is that blockchain is, at its core, a useful way of recording signed statements and agreements, and then providing a way of managing the lifecycle of those agreements.

This process is generally referred to as 'Dematerialization'[21] in the traditional financial world when applied to securities and refers to the use of ledgers and book-keeping to manage ownership of securities.

We use the word 'Tokenization' to refer to the broader use of encapsulating these relationships and agreements into usable assets that can be issued, transacted, settled, and modified using blockchain and related technologies.

This tokenization can refer to much more than just the representation of assets and cash; indeed, we can 'tokenize' many types of information, including identity and contracts.

---

[21] https://en.wikipedia.org/wiki/Dematerialization_(securities)

This tokenized data can then be used, reused, and transferred across many different systems and networks, all the while pointing back to the initial source and provenance of the data, building trust that it is indeed a good, verified piece of data.

Centralization vs Federation vs Decentralization

Much of the focus of the discussions surrounding blockchain, and especially cryptocurrencies, has centered around centralization versus decentralization.

The simplified discussion usually points Bitcoin (or other cryptocurrencies) as being globally decentralized and permissionless, and any system in the financial sector as being centralized and closed. As with any good oversimplification, this is superficially correct but at the same time, misses the point entirely. When you take a look behind the curtain, it becomes quite obvious that both views are based in a utopian mindset, where everything slots into perfect definitions.

In reality, Bitcoin is much more centralized, fragile, and permissioned than general perception, while the finance industry is much less centralized and monolithic than generally given credit for in these discussions. While any given financial institute is generally a centralized entity, they act in a decentralized manner when it comes time to transact between themselves.

As such, they need to find an intermediary that can be trusted to settle transactions between themselves. In any ledger, we're going to be concerned that there is one, canonical truth for all interactions between participants.

Traditionally, we've accomplished this by marking one organization's ledger as the source of truth in a hub-and-spoke manner.
This has resulted in the creation of entities like Central Securities Depositories, which financial institutions depend on to be the ultimate source of truth for their position at any given time.

However, DLT gives us the ability for us to create a canonical source of truth for a ledger without needing a third-party to act as the System of Record[22] for their ledger entries. Instead, the actions taken by participants in the ledger create that truth in a distributed fashion, according to the consensus rules of the network.

As technologies are developed which allow for transactions to span across network boundaries, we can take a natively federated approach to onboarding identity, value, and other information to these distributed ledgers.
This allows for tokenized information that was built on one network, whether centralized or decentralized, to be used in many different contexts, rather than only having value in one organization's ledger at any given time.

---

[22] https://en.wikipedia.org/wiki/System_of_record

P2P vs Decentralization

Two topics that have been inextricably linked in the public eye have been Peer-to-Peer (P2P) and decentralization of services, especially since the initial hype of Bitcoin & Blockchain.
On some level, this makes a lot of sense; it is indeed true that in many use cases, making them P2P does primarily make the participants more decentralized and, in some cases, the platform they use more decentralized.

However, most use cases that we have seen take off where P2P is the unique selling point, the platform is still highly centralized, and this is not necessarily a bad thing.
However, P2P and decentralization are orthogonal to each other, and each can be utilized in isolation.

You can have any combination of the following:

- P2P & Decentralized – This is the idealistic view of technologies such as Bitcoin, BitTorrent, and Ethereum. The vision is that the service itself is distributed widely enough between users that it's difficult for any one party to shut down or compromise the service, but also that each user directly accesses the underlying infrastructure.
- P2P & Centralized – Services in this category rely on a centralized platform but allow users to directly connect to and procure the service. We have seen a multitude of companies attract much attention in the past few years under this model; from P2P lending to P2P foreign exchange transfers. In these cases, the improvement in efficiency comes from directly connecting each side of a two-sided market, allowing for easy discovery through a central marketplace.
- Not P2P & Decentralized – One prominent example of this would be any hawalla[23] payments network. While the intermediaries are relatively decentralized, all end users must go through these intermediaries to effect a payment.
- Not P2P & Centralized – Our current payments system would be an example of this. Money is issued by a central bank, and users are not able to directly gain access to central bank money; instead, they need to go through a small number of intermediaries who are able to settle central bank money on their behalf.

Of course, these are not binary properties; how decentralized or P2P a service is in practice is a sliding scale, not an absolute. It's also not clear there is necessarily a reason to favor one combination over another in general, rather looking at each service in isolation to determine the needs of that particular use case.
Rather than being ideologically wedded to one end of these two spectrums, it's important to realize when each of these models may be useful.

---

[23] https://en.wikipedia.org/wiki/Hawala

For example, the P2P model may be less competitive than a model where an intermediary is able to pool risk and operate at economies of scale, much like we've seen in lending and insurance. However, there are likely business models where the intermediary tends to take more in fees than they provide in efficiency, making the service more expensive than it needs to be, much as we've been seeing in the retail foreign exchange market.

Blockchain as a Horizontal Technology Layer

PLUG was very much built to integrate well with existing technologies at all layers of the stack. As shown in the chart further in this document explaining our product positioning, we sit above the infrastructure layer and below the application & service layer, allowing those building end services to abstract away the difficulty of building a distributed, cryptographic system.

Much like you wouldn't see a web server or a database as a self-contained application, but rather part of a larger application stack, we largely view blockchain technology as being a distributed, cryptographic, identity-driven database that should be able to seamlessly integrate with existing infrastructure, services, and applications.

One source of friction with early frameworks that attempted to build a framework on a cryptocurrency codebase is that the cryptocurrencies have largely been built to be self-contained. As a result, there is little out-of-the-box support for reusable components and integrations, leaving these to be bolted on after the fact.

For example, many enterprise-focused use cases become readily feasible if existing, populated PKI systems can be utilized to bootstrap identity, roles, and value. In many of the cryptocurrency-based frameworks, this would require a major refactoring of much of the codebase. However, PLUG has shown the ability to both perform the integration with the data store and the ability to validate signatures and authentications against this data.

Blockchain Interoperability

To truly benefit from a rich ecosystem of blockchain-based services, we want the different use cases to be able to work with and build upon each other. This allows for complex ecosystems to develop and ensures we have the best shot at future-proofing initial implementations of this nascent technology.

There are three main approaches to enabling blockchain interoperability:

- Place all blockchain-enabled applications on a central, global blockchain and enable those applications to communicate with each other on-chain. This is the approach of the Ethereum cryptocurrency.

- Allow separate blockchains for separate features and use cases, but have delegated or federated custody of funds derived from a "primary" blockchain that can be temporarily parked in use case specific blockchains. This is the approach of Bitcoin Sidechains.
- Allow separate blockchains to interact directly; directly exposing the ability to affect multiple blockchains in one, atomic transaction. This is the approach of PLUG Crosschains.

While each of these three have their advantages and disadvantages, it is (naturally) our firm belief that the third option allows for the most flexibility, control, and security for those looking to build permissioned blockchains for business-driven blockchain services.

PoW vs Finality

While Proof of Work (PoW) mining has many[24], well-understood weaknesses as a consensus algorithm, one of the issues that is not often discussed is the lack of finality in any PoW. Mining is often seen as the key factor in making cryptocurrency-based blockchains immutable, but in fact this is only the case in a simplified sense and, in practice, largely relies on altruism.[25]

This immutability is referred to as finality in the financial world and refers to the concept of the time when a given transaction or ledger entry cannot be reversed. Under PoW mining, convention says to wait anywhere up to six blocks (an hour on average) to assume a Bitcoin transaction is finalized, but there have been exceptional[26] times[27] where the Bitcoin blockchain has been rolled back by multiple hours due to bugs. This would look no different under a malicious attack on a PoW-based blockchain as there is no condition to enforce finality that is able to be enforced in PoW.

In fact, PoW mining explicitly allows anyone with >50% of the hashing power to arbitrarily rewrite history; it simply relies on game theory assumptions to avoid incentivizing bad actors. These assumptions may indeed hold under most single-chain, single-token use cases, but we have very little research on how these assumptions will hold up in a multi-chain, multi-asset system. Proposals for inter-blockchain functionality account for this using workarounds such as waiting a day and a half before assuming finality. [28]

However, using the leaderless variants of traditional consensus algorithms, as PLUG does with our distributed two-phase commit, allows for true finality in a relatively short period of time. This is an especially important attribute to consider when enabling inter-blockchain features, as we would not want to act on information or events happening in an outside blockchain network if we could not rely on that information or events to be rolled back arbitrarily in the future.

---

[24] https://news.ycombinator.com/item?id=10774773

[25] https://blog.ethereum.org/2015/01/28/p-epsilon-attack/

[26] https://bitcoinmagazine.com/articles/bitcoin-network-shaken-by-blockchain-fork-1363144448/

[27] https://en.bitcoin.it/wiki/Common_Vulnerabilities_and_Exposures#CVE-2010-5139

[28] https://gendal.me/2014/10/26/a-simple-explanation-of-bitcoin-sidechains/

Inter-Network Atomicity & Delivery vs Payment

One of the areas of greatest potential within financial services is the ability to remove execution risk from counterparty transactions that involve assets where the end custody is maintained by disparate entities.

The challenge of ensuring both sides deliver against their agreement is generally referred to as "Delivery vs Payment" or "Payment vs Payment" and refers to whether a system can guarantee payment is not delivered until the asset is received, and vice versa.

As traditional ledgers are generally unable to intrinsically guarantee this is met, we assign an entity such as a Central (Clearing) Counterparty (CCP) to hold the risk of settlement once both parties have come to an agreement.
The CCP then holds the liability and risk to deliver both sides of the deal.
This risk arises from a few primary issues:

- The custody of the asset class is determined by a third party, often dependent on jurisdiction and asset class.
- Finality of an asset transfer often takes a time, and the length of this period depends on asset class, jurisdiction, and transfer method.
- The length of the finality period can differ between the separate sides of a transaction.

As we can make certain guarantees about the atomicity of a Crosschain transaction, the execution risk of either side of the agreement disappears. Either delivery happens on all sides of an agreement, or the agreement is completely rolled back. Due to the fast guarantees we can make on finality, this also reduces the opportunity cost of tying up assets during a prolonged settlement period, instead making the assets available almost immediately.

Token Custody vs Functional Interoperability

While most of the implementations for interoperable blockchains have concentrated on allowing for delegated custody of tokens, sidechains being the prime example, there is also the ability to have execution and functionality of a given use case to be split across multiple blockchains.

Simple delegated custody looks much like a hub-and-spoke model of custody, where assets are parked in a blockchain other than the one they were created in, any use, trading or movement is done in this third party blockchain, then the assets are moved back into the original blockchain, thus freeing them up for further use.
We instead propose a method of allowing for transactions and other functionality to reference multiple blockchains that a single transaction should be executed against.

For example:

1. UserA holds 10 USD in BlockchainA and wants to trade for 10 shares of stock held by UserB in BlockchainB.
2. UserA prepares and signs a transaction that references both BlockchainA and BlockchainB, specifying the intended trade, then passes it to UserB.
3. UserB signs the prepared transaction and submits it to both blockchains.
4. Each blockchain pre-commits processing its end of the transaction upon transaction confirmation. In this case, 10 USD would be debited from UserA in BlockchainA and 10 shares of stock would be debited from UserB in BlockchainB.
5. Once each blockchain sees all other blockchains in the transaction have confirmed the transaction, the transaction is finalized. In this example, 10 USD is credited to UserB in BlockchainA and 10 shares of stock are credited to UserA in BlockchainB, thus completing the transfer without any execution risk.
6. 6. In the event it becomes impossible for the other blockchain to confirm the transaction, a rollback is executed to undo the half-committed change from Step 4.

This effectively utilizes a multi-network two phase commit for any transactions that depend on changing state in two blockchains in a transaction.

Rather than requiring that value is first moved into a third party blockchain, the change is coordinated directly between the two blockchains in an atomic fashion, removing the tradeoff between execution risk and opportunity cost associated with collateral management.

Passporting of Tokenized Data

Of course, custody of tokenized information can refer to much more than just currency or assets. In fact, much of the time we will want to reuse information that is completely non-financial in nature, especially regarding identity data.

"Passporting" of this data allows the data to be used natively in completely independent blockchain networks, one example being federated proofs of identity, allowing provenance trails to move with the data, wrapping the authoritative proof of data source around the underlying data itself.

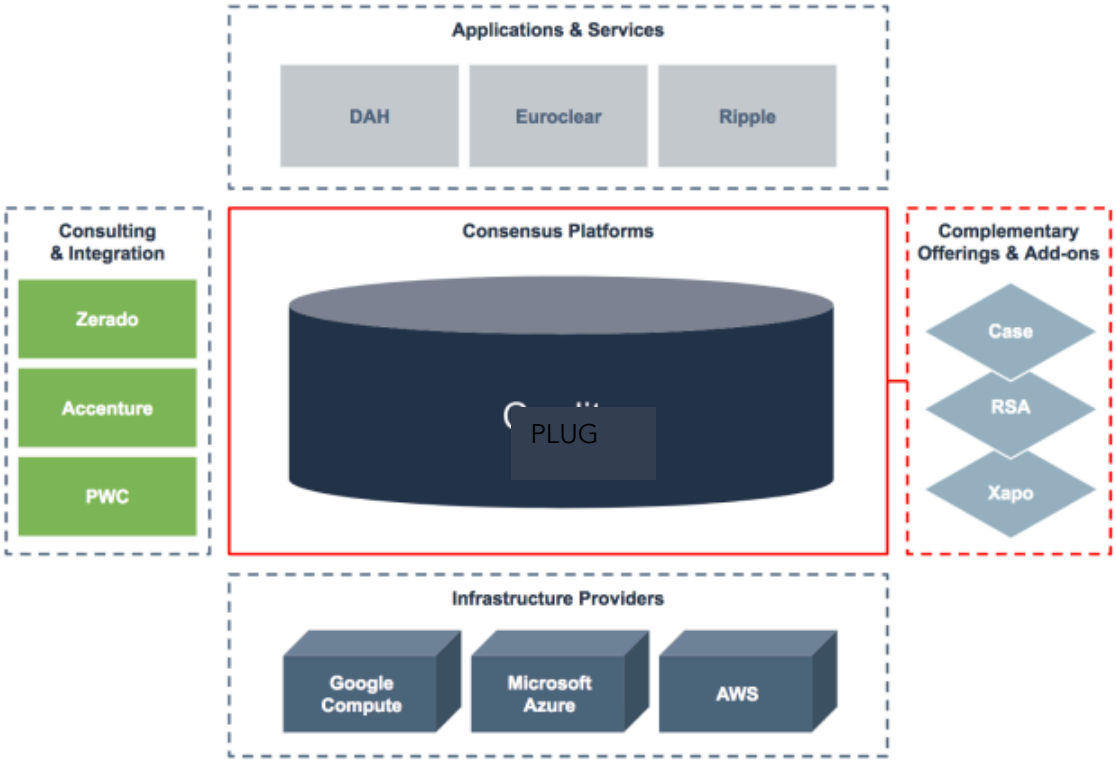
PLUG Products & Technology

The technology that has been created by PLUG[29] can largely be grouped into the PLUG Core and the PLUG Platform.
The PLUG Framework provides the ability to stand up permissioned blockchain nodes that can be assembled into a fully-functioning blockchain network, while the PLUG Platform provides tooling and services that enable the deployment and maintenance of blockchain networks on arbitrary hardware.

[29] http://credits.readthedocs.io/en/latest/index.html

Technology is very specialized today; just as you would generally not want an in-house team to build a database from scratch to build a database-driven application, you should generally shy away from requiring an in-house team to build a consensus, blockchain, or DLT framework for your specific use case.

PLUG sits in the middle of the Infrastructure and Application layer and allows organizations to use cryptography and distributed consensus to manage identity, enforce business logic, and create tamper-proof data; all without needing to implement these features from scratch in a bespoke, one-off manner.



Largely speaking, we see a group of primary features that blockchain provides which businesses are increasingly finding valuable to incorporate into their services:

- Distributed, Master-Master Infrastructure
- Public Key Cryptography for Authentication, rather than Usernames & Passwords
- Inherent Failover and Uptime Guarantees
- Strong, Declarative RBAC Configurations
- Enforcement of Business Logic at Protocol Layer
- Tamper-Proof Audit Trails
- Zero-Downtime Migrations
- Tokenization of Data, Identity, and Value
- Horizontal *Read* Scaling (today)
- Horizontal *Write* Scaling (R&D Stage)
- Inter-Network Atomic Transactions (R&D Stage)

It is true that for any small subset of these features, it's likely not insurmountable to implement any subset of these features in a one-off or proprietary fashion. After all, like most infrastructures and platforms, any given use case tends to use at most 20% of the underlying features available.

It's also almost assuredly the case you would spend more time on initial implementation of these features in your application layer, not even counting maintenance and upkeep, than you would spend in licensing or upskilling on a framework, such as PLUG.

Having said this, at PLUG we strive to ensure our technology truly acts as a horizontal technology layer, integrating well with existing layers of the technology stack. We have built it in a truly modular fashion, allowing you to pick and choose which layers are relevant for your use case, both present and future.

The PLUG Framework

The PLUG Framework is a library that allows an individual or organization to build blockchain-based applications or services without needing to implement the underlying blockchain technologies themselves.

The framework also allows for frictionless integrations with legacy systems and plugging into the 'plumbing' that already runs most of the systems meant to be augmented by blockchain, in the enterprise.

PLUG is currently written in python, although there are plans, expanded upon in the Project Theseus section, to allow for development against arbitrary programming languages. In the meantime, we are in the process of building support for arbitrary, hosted VMs in user land that may allow for functionality to be built in any number of Domain Specific Languages (DSLs) or traditional programming and scripting languages.

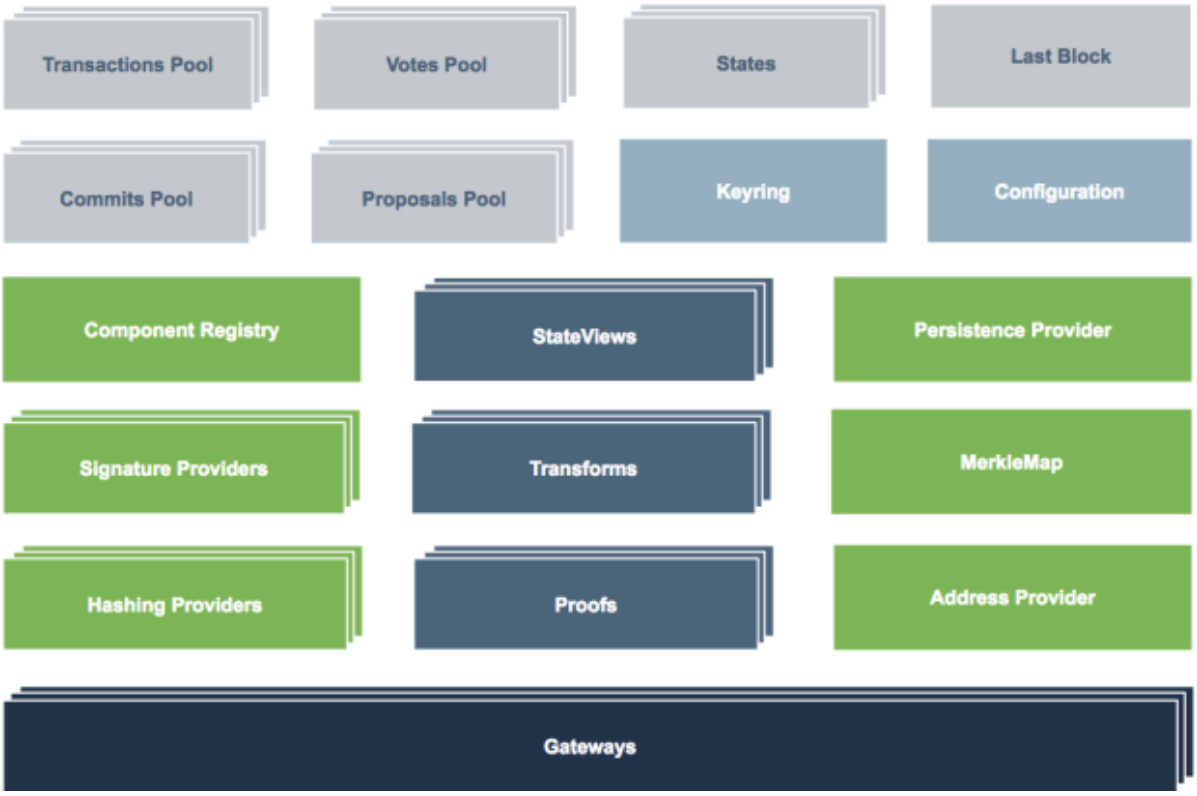
When explaining features and the state of these features below, I will generally refer to whether these are features that are:

- Available today.
- Something we consider trivial and we will implement once customer demand prioritizes them.
- Something we know how to execute on, but may take a few engineer-months.
- Something that is in the R&D stage, but looks doable.
- Something that is more speculative, but makes sense conceptually.

Modular, Extensible Design

The PLUG Framework was built from the ground up in order to be incredibly flexible, modular, and extensible. Think of it as if you were being given a box of Lego you could assemble in any which way, as opposed to a model airplane that can only be assembled in one manner. As a result, it is straightforward to pull any given module in the system for use in new or existing products.
It is also straightforward to enable or disable functionality in any given network, depending on the needs of those running the network.

You can see below a general overview of how the different components in our framework are laid out. The boxes which show an overlap are the components where you can enable an arbitrary number of components in any given network.



Largely speaking, the system is divided up in that the grey boxes are consensus primitives, the green boxes are cryptographic and systems integration primitives, and the blue boxes are user land.

As discussed elsewhere in this document, we generally recommend users stick to implementing their functionality in the components represented in the blue boxes, but we natively expose the ability to modify any of the components in the system, up to and including the consensus algorithm.

Consensus Mechanism

While consensus is configurable to each Plug network, PLUG Core consensus is a leaderless two-phase commit algorithm with variable voting power assigned to each 'Validator' node; or 'Miner' in cryptocurrency terms.

Each DLT network participant is equal in its rights to gather transactions from the unconfirmed transactions pool and form a block, and each is free to vote on the block candidate for the next round that make the most sense according to current block validation rules.

The first phase of the two-phase commit involves a 'pre-commit' where each node votes for the candidate block. Once a quorum of participants have pre-committed to a block, nodes race to finalize their commitment to the block the network has chosen, thus locking in that block as the canonical block for that height.

This candidate block may be proposed by anyone with valid transactions which validate against the current state, however in practice nodes may use meta-protocol heuristics in order to avoid validating too many contentious blocks per commit round.

During the consensus process, votes may have different weights according to voting power distribution which is set for a given network in the State of that network.

This mechanism differs slightly from a traditional two-phase commit in the following ways:

- Allowing consensus on a quorum of validators reaching consensus, rather than 100% of validators.
- Allowing validators to have differently weighted influence on the consensus process, dependent on their staked value.
- As this is a blockchain, coming to consensus on a block of transactions rather than one transaction at a time. A traditional two-phase commit requires 3n messages per n nodes per transaction, but pooling transactions in a block allows us to amortize this over a large number of transactions.
- We allow the membership of the validator pool to change over time as validators stake or unstake their token, as staked validators become inactive and are automatically unstaked, or if a staked validator breaks the consensus mechanism and is forcibly removed from the pool.

In cryptocurrency terms, PLUG Core consensus is a variant of a Proof of Stake algorithm.

Several main benefits have manifested from this consensus algorithm in most use cases we have explored with clients:

- True finality is reached after each block. Immediately.
- The consensus algorithm is lightweight and does not require the tying up of unrelated computing resources for security.
- As a leaderless protocol, this consensus mechanism does not suffer from the scalability issues and ceiling that leader-based consensus protocols intrinsically suffer from.
- The consensus algorithm is very flexible in modeling real-world relationships and allowing any use case to choose exactly how centralized or decentralized their use case requires their blockchain network to be.

For more detailed information, visit our developer docs.[30]

There is a chance we will replace this consensus mechanism for Plugnet by the time we launch the public Plugnet, and we certainly will have multiple consensus algorithms available for those creating Plug-based blockchains.

One major benefit to our modular design has been in our ability to support migrations of existing data and integrations with existing systems.
One common requirement that cryptocurrency-based blockchain platforms often are unable to meet is the bootstrapping of identity and credentials using an existing internal CA or internal PKI System.

[30] http://credits.readthedocs.io/en/latest/blockchain_details.html#credits-core-consensus

Since we have built a very agnostic framework that allows for the plugging in of many different types of core integrations, we have managed to make these integrations virtually seamless, allowing teams to concentrate on their domain and technical expertise, rather than wasting precious time fighting their tools and existing systems.

User Land

We generally refer to a few key components that are intrinsic to implementing business logic and integration with existing systems as user land.
When looking at which modular pieces from our Core architecture we have designated as user land, it's important to note that there is not a black and white distinction between the parts that a user cannot touch, rather we have identified a few components that will make sense for most users to modify to accomplish their goals.

You are still free to modify any of the other components of the system, and we do expose these to end users, but it's unlikely that for most use cases there will be a need to modify anything outside of 'user land' components, or that the benefit will exceed the risk and cost of doing so.

States & Models

The goal of any blockchain is to assemble an agreed upon state of the world in a distributed manner; the PLUG Framework, by default, exposes a key/value store that allows any string as a key and any primitive or primitive collection (Map, List) of primitives to be the value.

This data is then stored as a Merkle Tree, allowing for very granular verification of individual data elements and tamper-proofing of any state stored in the blockchain.

Models allow a default value to be passed as well, simplifying the business logic for many use cases. For example, if you're storing token balances in a State, you may want the default balance to be 0, rather than adding complexity in ensuring there is a balance present before performing any action on the State and removing surface area in your code for bugs.

In the future, you will also be able to enforce schemas in Models, much like you would define a schema in SQL or an ORM. This will allow you to encode business logic constraints at the data layer which will usually remove certain classes of bugs that can happen higher in the application stack.

Transforms

Transforms take a user-defined payload and act upon a State object to modify it. This can be thought of as an authenticated API call.
Transforms define business logic that validates the user-defined payload and verifies it is valid against the current State (or States) of the world. Assuming the Transform properly validates against

State, the Transform additionally provides business logic for modifying State using the user-defined payload.
There are a few methods and properties a Transform must implement in order to be valid:

- **Verify** – Business logic that a given payload is valid against the current State.
- **Apply** – Business logic to modify State utilizing a user-defined payload.
- **Required_models** – Any States the Transform needs to be able to access to perform its verification and application.
- **Required_key** – Any Keys within a State a Transform needs access to perform its verification and application.
- **Required_authorizations** – Any users that are required to present a cryptographic proof that they are indeed authorized to make the change laid out in the payload and Transform.
- **Marshall** – A method that defines the marshalling out of the complex Transform class to a plain dictionary. *This boilerplate will be removed soon.*
- **Unmarshall** – A method that defines the un-marshalling of a plain dictionary into a complex Transform class. *This boilerplate will be removed soon.*

It's worth noting that a Transform only concerns itself with the specific business logic of a use case and does not include logic for either role-based permissioning or individual permissioning. This is implemented in Proofs for individual permissions and as part of the RBAC State for role-based permissioning.

Proofs

Proofs represent an authentication for a Transform and consists of a payload that can be cryptographically validated against an address or identity living in a PLUG blockchain.
This identity can represent one or multiple public/private key pairs, can represent a pseudonymous or real-world identity, and can be machine-readable or human-readable. All configuration as to how this identity and authentication are performed is able to be user-defined; although we do recommend that users generally stick to PLUG-blessed Proof types.

This is separate from authorization, which is performed jointly by the business logic implemented in any Transform's verify method in conjunction with the run-time RBAC configuration.
One or more Proofs are paired up with a Transform to form a full Transaction.
Proofs are an Applicable type.

Transactions

Transactions take exactly one Transform and pair it up with one or more Proofs which serve as authorization for that Transform and the associated payload.
A Block is made up of one or more Transactions that are executed in order; in general, a blockchain bundles up multiple Transactions for consensus as an optimization, rather than confirming each Transaction separately.

Transactions, like all Applicables in the PLUG Framework, are confirmed in an ACID[31] and atomic manner.

Blocks

Largely speaking, blocks are a wrapper around a list of Transactions, paired with some metadata that ensures the system is coming to the right consensus around the history of the blockchain.
Blocks are proposed by any validator node, and a block for each height is agreed upon in a decentralized manner, according to the consensus protocol described above.

Gateways

Gateways are simple conceptually in that they take complex PLUG objects, serialize them out into bytes, then punch them across the wire in some capacity.
Gateways can be truly peer-to-peer and pass full consensus data as it's generated, or as they see it; or they can act in a client-server capacity, giving responses to client requests or pushing notifications to clients which have subscribed to such notifications.

By default, we ship a ZMQ and HTTP(s) gateway for P2P and client-server communications, respectively. A websocket gateway is coming soon, and we are working with early partners on domain-specific gateways, such as a SWIFT messaging gateway.
An arbitrary number of gateways can be enabled on any given network, allowing for consensus to happen, even if a quorum of validator nodes cannot communicate over the same protocol channels.

Cryptographic Primitives

By default, we ship with SHA256 as our hash mechanism and ed25519 as our signature scheme. This is trivially customizable, and most common cryptographic primitives take between 20 and 50 lines of code to implement.
You can, of course, enable as many or as few cryptographic primitives to be available to the system for any PLUG blockchain network. Most of the larger organizations that have been working with the framework are particularly interested in being able to support legacy PKI systems that generally require specific signature and padding schemes, support for which is easily achievable within the PLUG Framework.

Abstract Interfaces

We have four main interfaces that many of the components in the PLUG Framework comply with. These interfaces are described below and form the building blocks for many of the PLUG Core primitives.

---

[31] https://en.wikipedia.org/wiki/ACID

Applicables

An Applicable is a class that can be verified and applied and implements the verify and apply methods. Classes that inherit from this class should be able to be verified against a state, which acts like a dictionary of default dictionaries, and should be able to apply against the same state, updating any values their payloads indicate should be updated. This class primarily serves as the basis for Transforms, Proofs, Transactions, and Blocks.

Marshallables

A Marshallable is a class that can export, or marshal, its contents to a dictionary as well as import, or unmarshal, a plain dictionary into a complex, PLUG object. Any component that needs to be passed around the network will generally inherit from the Marshallable class.

Signable

A Signable is a class that is meant to be signed by a participant, or participants, in a system, and is then verified by other users in the network. This class primarily serves as the basis for Proofs.

Hashable

A Hashable is a class that exposes the hash method and has a way of representing the contents of the class uniquely as a hash string. Often the marshalled output is passed through a custom digest function and the resulting bytes from that are hashed, ensuring all relevant data feeds into the entropy input for the hash in a deterministic manner.

Merkle Tree Data

Merkle Trees[32] are a conceptually simple, but very powerful cryptographic data structure. Effectively, they allow you to efficiently verify the cryptographic integrity of any individual piece of data within a larger data structure.

While most blockchain platforms to date only allow for lists and maps to be placed into a Merkle Tree, you can trivially convert most types of data to a Merkle Tree format; in theory, allowing you to seamlessly upgrade most existing ways of storing data into a format that can be validated and managed by a blockchain-like or DLT-like structure.

This is also the data structure that enables Simplified Protocol Verification, embedded device integration, Crosschain validation, and tokenization of data.

---

[32] https://en.wikipedia.org/wiki/Merkle_tree

Time-Based Events

Time-based events allow you to perform 'Smart Contract' style execution of logic, based on the time in the real-world, as asserted collectively by the Validator Network. This allows you to do things such as set max thresholds for payments sent per day/week/month, potentially in line with your KYC/AML requirements. You can also schedule more complex interactions such as auctions and expiring derivatives.

Simplified Protocol Verifications (SPV Proofs)

Simplified Protocol Verification, or SPV, is simply a strategy of using Merkle Tree proofs and consensus signatures to verify the entire State of a blockchain by only specifically verifying a small portion of the total history. This technique involves making sure each block reported by the network nodes does indeed contain a quorum of the signatures necessary for it to be considered a valid, permanent block.

If this, combined with the State & Block headers, continues to validate, it's generally safe to assume the network validators are not surreptitiously modifying historic or current state. Thus, you can rely on any data coming back from the blockchain that also may be depended on for your use case. It's worth noting this is also a significantly important building block for building Crosschains, PLUG' cross-blockchain interoperability product. Combined with some clever cryptography, this approach allows us to verify and send data in a compact format that can be used decisively in other, connected blockchains.

This technique also is critical for any IoT and embedded use cases that may be able to take advantage of blockchain.

Role-Based Access Control (RBAC)

Role-Based Access Control (RBAC) is provided out of the box, declaratively through configuration. Utilizing RBAC does not require that you conflate your business logic and your authorization logic, rather you define your business logic in a PLUG module and then separately define roles and membership in your network-specific configuration.

This allows for non-developers to securely manage roles and permissions in an organization's systems and blockchain networks, also ensuring that you can change these roles and permissions in a running system, rather than requiring a network functionality upgrade to change the way one permissions actions on a network.

It also enforces a very clean separation of concerns, leading to much safer and more secure coding and deployment practices; a paramount concern in blockchain networks.

This also has the benefit of being able to bootstrap roles and permissions from a legacy database or application, rather than needing to recreate your organizations permissions from scratch when deploying a blockchain-enabled service.

Privacy

Unlike most blockchains, PLUG allows for data in the state to be kept private from users who are not authorized to view it. Depending on the exact privacy mechanism used, permissions on read access to data can be declaratively set as part of the RBAC configuration of a network or rely on key sharing which is managed by users at the meta-protocol layer.

There are four main methods of managing privacy which we recommend:

- Encrypting data at the application layer and using a messaging or meta-protocol layer to share decryption keys to those users who you wish to allow to see the underlying data.
- Only storing a pointer to data that is stored elsewhere, generally in a traditional or domain-specific information store or database, and then using traditional access control to enforce who can see what data.
- Storing the underlying data directly in the blockchain, allowing the full dataset to be viewed by the Validator Nodes on the network, but using the RBAC configuration of the network, in conjunction with permissions middleware, to strip any unauthorized data from any responses going to users of the system. This can be bootstrapped with existing PKI systems or internal CA.
- Using the previous strategy, but using Crosschain functionality to connect many different blockchains that share data with the individual participants, but hide private details when syncing up to or netting off against the connected blockchains in the larger, logical network.

Unlike cryptocurrency networks, most permissioned blockchain networks will likely require a privacy component. This is especially true given the fact that real-world identity will often underpin these use cases, meaning stringent precautions against leaking sensitive user data must be taken, both for legal and competitive reasons.

PLUG CLI

The PLUG CLI allows for much easier management of the development, testing, staging, and deployment environment of PLUG modules and PLUG networks.

During development, you can run tests, create & manage local test networks, and manage keyrings for credentials and authentication. This tooling is in the process of being built out to allow for arbitrarily complex network management, including the ability to deploy and manage networks which are made up of multiple types of nodes, transfer voting power from old nodes to new nodes, and bootstrap the generation of a network in the first place.

MapReduce

It's all very well and good that data is maintained in a tamper-proof, auditable fashion in most every blockchain framework in existence; however, this data is only useful if you can pull it back out of the blockchain in a useful fashion.

Complex queries against blockchains often requires third-party tooling which exports the underlying blockchain data to a format more suitable for querying, such as a SQL database or traditional document store, rather than allowing complex queries against state and history natively.

PLUG instead allows you to construct arbitrarily complex queries programmatically that query against current state, past state, past blocks, or transaction history. These queries can currently be written in Lua, Python, or Javascript; but there is no reason arbitrary programming languages cannot be supported, assuming an embedded VM for that language can be hosted in our runtime.

For queries that are run frequently, the overhead associated with MapReduce (O(n) Complexity (Simplification, but approximately correct for what we're measuring here. Embedded VMs & DSLs)) may prove cumbersome. To combat this, we expose the ability to 'index' a query, allowing for instant query results at the expense of a small amount of overhead when processing a block.

There are two approaches we are taking to allowing for blockchain functionality to be written in arbitrary traditional programming languages as well as Domain Specific Languages (DSLs). These approaches are meant to allow much greater flexibility and portability in the first instance and much greater safety and productivity in the second.

While we currently only directly support the building of User land objects in Python, the same work we have completed to integrate MapReduce queries in several embedded VMs allows us to expose the building of PLUG modules in multiple programming languages; again, including Lua and Javascript.

We are currently waiting on a few simplifications of the Marshallable interface before exposing and documenting this functionality in greater detail, but the underlying capability is now present.

We have additionally incorporated DSLs natively into PLUG modules in the past, and one notable example of this is by wiring in a standardized chess DSL, allowing for a full chess engine to be running on the blockchain. More practical DSLs will include capabilities like FIX protocol and other financial and industry-driven DSLs, allowing both technical and non-technical staff to add domain-specific functionality to a running blockchain in a safe, secure manner that incorporates their intent in a straightforward manner.

Project Theseus

Project Theseus is named after the Ship of Theseus[33], in that we intend to rewrite each constituent component of the PLUG Framework in a low-level programming language, replacing each part until we have a functionally equivalent framework that is merely wrapped and exposed in python.
This will allow us to expose the PLUG Framework in any modern, high-level language, expanding the skillsets that are able to harness the power of blockchains and integrate any given PLUG solution into their existing code bases, services, and work flows.

This will likely be most useful in conjunction with functionality that is primarily driven by DSLs, rather than native code; an additional reason for choosing DSLs over a 'Turing-complete' language for blockchain functionality, over and above simplicity and security.

Scalability Roadmap

In a previous generation of the architecture, we proved the ability to process approximately 136,000 Transactions per Second (TPS) on a small network of 500 GBP workstations.

This is also the power that has allowed some of our customers to be able to credibly claim scalability over and above 100,000 TPS.

On our current architecture, we are currently in the high 4-figures (X,000s) and low 5-figures (X0,000s) TPS range as we have devoted little time to optimization, concentrating instead on building out feature sets.

We have deferred prioritizing throughput optimization until we see use cases that could presumably take advantage of scalability beyond tens of thousands of transactions per second; however, we have a scalability roadmap that should bring us quickly back to our previous figures and above based on the results we have been seeing at the R&D level.
We consider much of this to be a matter of prioritizing engineer time to these problems, rather than risk associated with the concepts not working in practice.

PLUG Federated Clusters

One straightforward way to scale this workload is to divide it into the cryptographic processing and logic processing tasks.

Any of the checks on cryptographic integrity (hash checks, signature validation, Merkle Trees, etc.) are trivially an embarrassingly parallel problem.[34] This means, it's simply a matter of throwing money, hardware, servers at the problem, and we can (in theory) make throughput as high as we wish for this set of our workload.

---

[33] https://en.wikipedia.org/wiki/Ship_of_Theseus
[34] https://en.wikipedia.org/wiki/Embarrassingly_parallel

However, we do have an intrinsic bottleneck in verifying business logic, as there is often a dependency tree on which order transactions need to be verified. For example, if I have $20 in my bank account, but send three transactions of $10, there needs to be a way of processing them in order so I don't send more money than I have available.

In benchmarks we have run, this part of the workload should still have a throughput of over half a million transactions per second on a modern, commodity laptop; meaning this is the current, theoretical upper-bound of the framework if we were only using this scaling technique. Thankfully, we have several other approaches that complement this strategy well.

Dependency Trees

At a core level, the technical advantage a blockchain provides is the ability to order and timestamp signed messages in a distributed fashion. We build use cases on top of that which consider that order and depend upon that order to function, so we need to be able to rely on that order when implementing most use cases, especially in the financial world.

This means that for most types of validation, a naïve blockchain implementation must process each transaction in serial, that is one after another. This leads to an expectation that you would not be able to take advantage of modern, multi-core processors to be able to truly parallelize any blockchain implementation.

However, if we inspect each transaction and work out which transactions are dependent on each other, it turns out we can parallelize most workloads fairly trivially. This, of course, relies on us being able to automatically ascertain which models and specific records any given Transaction is requesting access to potentially update and ensuring that we don't process two Transactions in parallel that are dependent on each other. This is similar to row-level locking or per-record locking in a SQL DB or a Document Store.

For low-contention workloads, this allows for a workload that can be processed in a much more parallel fashion in practice, while still maintaining our guarantee of serialized ordering of transactions at a global level.

As a side note, the API for enabling this feature is already in place and necessary for the current implementation of any PLUG functionality. So, when we release this as a feature into production, you will get the scalability benefits for free, assuming you update your Core version.

Bulk Signature Validation

There are a few great properties that our default signature scheme (ed25519) holds. Among those are deterministic signatures, enhanced safety, and over 10x scalability over ECDSA. However, one interesting property of the specific curve is that it's also possible to validate more

than one signature at once, thus speeding up the cryptographic validation process by up to a factor of 64x.

As signature validation is currently our largest bottleneck, this one feature will go a long way towards removing much of the computational time we currently spend validating incoming messages, and thus speeding up significantly the processing of transactions.

The current state of this feature is that it is well documented how to achieve it, but we still do not have a practical implementation of this which is widely available. We hope to be able to sponsor those from both academia and industry in the near-to-medium future to make sure this benefit is available to anyone in an easy-to-use form going forward.

Sharding

One relatively straightforward path to horizontal scalability is to shard the state of the network across multiple physical networks, while still maintaining a global, logical network.

This allows you to split the data you are storing and the transactions that you're processing across multiple nodes, allowing each node to only keep a subset of the state of the world, while still maintaining the integrity and validity of the total state.

As a result, as each validator node is only responsible for a subset of the total network traffic, you can increase the amount of transactions and state processed by a network by raising the number of partitions, or shards, we split the state across.

This scaling strategy takes advantage of the fact that we can use Crosschain atomic transactions to connect multiple networks together so they act as one logical network.

Embedded Devices

PLUG natively works on many System on Chip (SOC) systems without modification. We have also demonstrated native integration of embedded chips and HSMs as users, or user authentication, of a PLUG network.

Work is planned to allow for a much wider variety of even smaller chips and devices as we start work on IoT integrations and toolkits, as well as native integration of SPV proofs, ensuring a device with limited capability does not have to process the entire history (or even current state) of the blockchain in order to use data from the blockchain in a secure, verifiable manner.

As we see the biggest advantage to blockchain as being a conduit between the real world and the computing world, it has been at the forefront of our focus to architect our framework in such a way that we present the ability to integrate natively with embedded systems while still retaining the ability to scale up to the largest and most powerful computers.

Internationalization (i18n)

PLUG is, at the time of publication, the only blockchain platform that allows for seamless i18n integration, allowing for messages, responses, and logs to be returned in your local language. A node running in France can return messages and keep logs in French while still keeping full consensus with nodes in other countries who message and log in English, German, Mandarin, and so on.

This is an example of one of the necessary requirements to be a robust, production system that can operate across borders, something we have been making sure we deliver on from the outset.

The PLUG Platform

Basic Overview

The PLUG Platform is a set of tooling for deploying and maintaining blockchain networks based on the PLUG Framework.

Our platform can be used with any major cloud provider or can alternatively be used with on premise equipment, depending on the requirements of the client. This platform takes configuration files and, depending on the use case either automatically or semi-automatically, uses the specified configuration to launch and maintain networks based on that specification.

As topology and configurations are specified declaratively, this allows for truly deterministic builds, removing much of the risk and many of the edge cases that manifest in traditional systems deployment.

Node Management

The PLUG Platform can monitor and maintain nodes that are running in any given network that has been deployed via its infrastructure.

If a user wants to migrate, update, or upgrade its nodes; this is achievable trivially by migrating to new servers and moving their voting power. One of the benefits of blockchain is that you can migrate a service slowly, node-by-node, and very easily never encounter any downtime while doing so.

Failover & Voting Power

As often happens, if a server does happen to go down, it's rather trivial to bring up a new server/node, sync it to the remaining nodes, then transfer voting power to the new server.

Much like an upgrade or migration, this can be done server-by-server and works seamlessly if the network never goes under a failure mode where a greater than quorum group of servers becomes irrecoverable at any given time.

Even so, in the permissioned blockchain environment, there are strategies that can be undertaken to make sure that you can recover, even from a truly catastrophic event where you have fewer than quorum servers left up and running.

CLI – PLUG Remote

The PLUG CLI allows developers to create and test blockchain functionality locally, as discussed in the previous section; however, we are slowly rolling out tools to generate and deploy networks remotely using the PLUG CLI.

The way the tools are being created also allows you to plug and play various network bootstrapping strategies based on the needs of those who are creating the network. For example, an internal system likely needs a different bootstrap and genesis verification process than does a consortium network.

The tools in the PLUG CLI for remote management will allow for these different strategies to be plugged in as required by those who are developing and deploying PLUG networks for their particular use cases and environments.

Bringing it all Together

To sum it up, we at PLUG firmly believe that this set of technologies will allow us to robustly onboard and manage real-world relationships in many different fields that we have failed to do so for to-date.

While PLUG has evolved far beyond its origins, you still see many of the original ideas shining through loud and clear. Most of the pivots in focus we have undertaken have been on the market, operational, and integration side, rather than undergoing much change in the core concepts that attracted us to this path in the first place.